AD737528

ALGORITHMS FOR MIX-MAX PROBLEMS IN HILBERT SPACES

by

Robert Wayne Hecht

This work was supported in part by the Joint Services Electronics Program (U. S. Army, U. S. Navy and U. S. Air Force) under Contract DAAB-07-67-C-0199, and in part by the Air Force Office of Scientific Research under Grant 68-1579.

Reproduction in whole or in part is permitted for any purpose of the United States Government.

This document has been approved for public release and sale; its distribution is unlimited.

Security Classification

DOCUMENT CONTROL DATA - R & D

(Security classification of title, body of abstract and indexing annotation must be entered when the overall report is classified)

1 ORIGINATING ACTIVITY *(Corporate author)*

Coordinated Science Laboratory
University of Illinois
Urbana, Illinois 61801

2a. REPORT SECURITY CLASSIFICATION

UNCLASSIFIED

2b. GROUP

3 REPORT TITLE

ALGORITHMS FOR MIN-MAX PROBLEMS IN HILBERT SPACES

4 DESCRIPTIVE NOTES *(Type of report and inclusive dates)*

5 AUTHOR(S) *(First name, middle initial, last name)*

Robert Wayne Hecht

6 REPORT DATE	7a. TOTAL NO. OF PAGES	7b. NO. OF REFS
January 1972	55	12

8a. CONTRACT OR GRANT NO.	9a. ORIGINATOR'S REPORT NUMBER(S)
DAAB-07-67-C-0199;	R-548
b. PROJECT NO. AFOSR Grant 68-1579	
c.	9b. OTHER REPORT NO(S) *(Any other numbers that may be assigned this report)* UILU-ENG 72-2207
d.	

10 DISTRIBUTION STATEMENT

This document has been approved for public release and sale; its distribution is unlimited.

11 SUPPLEMENTARY NOTES	12 SPONSORING MILITARY ACTIVITY
	Joint Services Electronics Program through U. S. Army Electronics Command; Air Force Office of Scientific Research

13 ABSTRACT

The problem considered is the minimization of a functional in Hilbert spaces, where the functional being considered is the maximum of a set of N functionals for each point in the Hilbert space. Two algorithms are presented. One is a gradient, or steepest-descent method. The other is a Newton-Raphson method. It is shown that the two algorithms are to be used together. The steepest-descent method is to be used first and then the Newton-Raphson method. To use the Newton-Raphson method, convexity is assumed. Both the theoretical and the numerical aspects of the algorithms are discussed.

DD FORM 1 NOV 65 1473

Security Classification

14 KEY WORDS	LINK A		LINK B		LINK C	
	ROLE	WT	ROLE	WT	ROLE	WT
Min-Max						
Algorithms						
Hilbert space						

ALGORITHMS FOR MIN-MAX PROBLEMS IN HILBERT SPACES

BY

ROBERT WAYNE HECHT
B.S., University of Illinois, 1965
M.S., University of Illinois, 1967

THESIS

Submitted in partial fulfillment of the requirements
for the degree of Doctor of Philosophy in Electrical Engineering
in the Graduate College of the
University of Illinois at Urbana-Champaign, 1972

Urbana, Illinois

ALGORITHMS FOR MIN-MAX PROBLEMS IN HILBERT SPACES

Robert Wayne Hecht, Ph.D.
Department of Electrical Engineering
University of Illinois at Urbana-Champaign, 1972

The problem considered is the minimization of a functional in Hilbert spaces, where the functional being considered is the maximum of a set of N functionals for each point in the Hilbert space. Two algorithms are presented. One is a gradient, or steepest-descent method. The other is a Newton-Raphson method. It is shown that the two algorithms are to be used together. The steepest-descent method is to be used first and then the Newton-Raphson method. To use the Newton-Raphson method, convexity is assumed. Both the theoretical and the numerical aspects of the algorithms are discussed.

ACKNOWLEDGEMENT

The author expresses sincere thanks to Professor W. R. Perkins, his advisor, for his guidance and continued encouragement during the course of this research. He also appreciates the excellent guidance from J. Medanic who helped him become involved in the solution of min-max problems. He further appreciates the discussions with Professor J. B. Cruz, Jr., and Professor P. Kokotovic. He expresses thanks to his wife Charlene for her constant understanding, encouragement and assistance during his graduate studies.

Finally the author wishes to acknowledge the support of the Coordinated Science Laboratory at the University of Illinois. Special thanks are due to Mrs. Carol Martin for her expert typing of this manuscript.

TABLE OF CONTENTS

1. INTRODUCTION

A method is presented which expands the class of min-max problems which can be solved using an algorithm programmable on a digital computer. The min-max problem deals with determining u^* and y^* such that $J(u^*,y^*) = \min_{u \in U} \max_{y \in Y}$ where U and Y are convex, bounded Hilbert spaces.

This type of problem can occur quite frequently when formulating mathematical models of physical systems. A quality of the system such as cost or position of some object can be represented as the variable J which we call a performance index.

Now J is a function of some variables represented by the vector u, which the designer can control or specify, but J is also a function of other variables y which cannot be specified and which are known only to lie within a given range. These variables may be unmeasurable and if they vary, it is assumed that the change is slow enough so that they may be considered time invariant while the control is being applied.

When faced with unknown variables, an optimally adaptive controller[1] which combines estimation and optimization could be used; however, parameter estimation may employ systems with high gains, which are usually susceptible to additive noise. An optimal controller in a noisy system may not keep the performance index near the minimum as in the noiseless case.

The difficulties involved in parameter estimation in the presence of noise leads one to other approaches to the probelm. One such approach is to find the control which minimizes the maximum value of the performance

index as a function of plant parameters.

There has been considerable theoretical development of the min-max problem[2,3,4] and the author is aware of four algorithms[5,6,7,8] which solve a min-max problem. These algorithms treat the minimizing variable u as a vector in a finite dimensional Euclidean space. The space Y is a finite set of points and y is an element from this space. The space Y can be considered a definite set of values which may be assumed by the unknown parameters.

This paper is concerned with expanding the algorithms of 6 and 8 to deal with the class of problems where the minimizing variable is a vector in a function space.

$$u(t) = \begin{bmatrix} u_1(t) \\ u_2(t) \\ \vdots \\ u_n(t) \end{bmatrix}$$

where $t_o \leq t \leq T$.

In other words, u(t) is the control which can change with time.

For example, consider the system with the state equation:

$$\dot{x} = Ax + Bu$$

and a performance index

$$J = x'(T)Qx(T) + \int_0^T u'(\tau)Hu(\tau)d\tau.$$

Assume H is a diagonal matrix and is not a function of time. Also, assume matrices A and B are constant over time, and assume that some of the elements in A are unknown, but lie within a given range. Furthermore, assume that each unknown element in A has K_ℓ values where K_ℓ denotes that the ℓth element has k values. Then if there are r such elements, we have a set of $\prod_{\ell=1}^{r} K_\ell$ points.

Let $N = \prod_{\ell=1}^{r} K_\ell$. Then the set of N points constitutes the maximizing space of the min-max problem

$$\min_{u} \max_{i \in 1,2 \ldots N} J_i(u)$$

where $J_i(u)$ is the performance index when A assumes the ith value in the set $(A_1, A_2, \ldots A_n)$.

Demjanov [4] developed a Taylor's series expansion of the max function for the Euclidean or parameter space problem. This expansion for Hilbert spaces is presented in Chapter 2. Also, from the theoretical development in his paper one can construct a gradient algorithm.

According to the algorithm by D. M. Salmon[5] one should start with a subspace V' consisting of two elements from the original max space. Then find the u* which solves the min-max problem consisting of the original min space and the new max space V'. With this u* find the element i' in the original max space which maximizes the performance index. Then add i' to V' and repeat. The stopping criteria will not be presented here; the reader is referred to the paper.

Finding the value of u* for the min-max problem with V' as the max space is naturally a min-max problem; therefore, this is not a complete algorithm.

The Newton-Raphson method by J. Medanic[8] could be used to solve this sub min-max problem. If the performance index is quadratic in the minimizing variable and the space V' is small, then the solution could be found in one step However, a combination of a gradient method, as the one presented by J. Heller[6], and the Newton-Raphson method will be more efficient than Salmon's method and and the Newton-Raphson method. The reason for this is that the space V' after several iterations will be larger than necessary, and because of this, the Newton-Raphson method will become very slow. For this reason we will not employ Salmon's algorithm.

The algorithm employing an elimination method by J. Medanic[7] theoretically cannot be used for solving problems in function spaces. In this algorithm one begins with a space bounded by hyperplanes. At an interior point of this space one can construct a hyperplane which divides the space into two parts. We are able to determine that the min-max does not exist on one of the two parts. We then can eliminate this half from the space being considered. One proceeds in this manner until the space that has not been eliminated consists of a very small volume.

The main difficulty with trying to use this algorithm to solve problems in function spaces is that it is impossible to enclose any region large or small with a finite number of hyperplanes. To enclose an area with an infinite number of hyperplanes would take an infinite amount of time.

Therefore, we will use a gradient method until we get close to the min-max and then use a second order method to reach the min-max.

Let us consider for a moment minimization problems where the function is continuous and smooth. The min-max problem does not fall in this class because, in general, the max function is not smooth. The Newton-Raphson method which we may alternately call the second order method or the quadratic method is useful only in regions in which the function is for practical purposes quadratic; that is, the function may not be quadratic in larger regions, but in a small region near the minimum it may be for all practical purposes quadratic and the quadratic method will be useful in this region. In problems where the function is not quadratic except near the minimum it is generally desirable to use some form of gradient method until you are near the minimum and then switch to the quadratic method. The quadratic method will give a direction and step size, but if the function is not quadratic, the step may be larger than one that decreases the function. An acceptable procedure then would be to use the direction found but decrease the step. However, since each iteration in a quadratic method is more complex than that of a gradient algorithm, it is more profitable to use a gradient method until one is near the minimum.

The min-max case is similar and a gradient method should be used before a quadratic method is used. However, even if each function J_i in (1.1) is quadratic and the number of points N in the max space is large (10 or more points can be considered large), then we will show that it is still advisable to use a gradient method first.

We will briefly look at what is contained in Chapters 2 through 7.

Chapter 2 is concerned with finding the first and second terms of the Taylor's series expansion of the max function in Hilbert spaces. This is necessary before developing a gradient and second order algorithm.

Chapter 3 deals with the basic proofs and methods necessary to develop a gradient algorithm in Hilbert spaces.

Chapter 4 presents the basic proofs and methods for the second order algorithm in Hilbert spaces.

Chapter 5 shows how to represent the gradients and other expressions derived in Chapters 3 and 4 in function spaces, a type of Hilbert space.

Chapter 6 presents some experimental results and discusses possible difficulties with the gradient and second order algorithms and discusses the problems encountered when the gradient algorithm was programmed.

Chapter 7 describes the detailed steps in constructing the gradient and second order algorithms for solving the min-max problem in function spaces.

2. TAYLOR'S SERIES EXPANSION OF THE MAX FUNCTION

Before we develop the first and second order algorithms, we must be able to find the first and second terms of the Taylor's series expansion of the max function.*

Now, even if the functions $J_i(x)$ are continuous and smooth, the max function $J(x) = \max_{i\in 1,\ldots N} J_i(x)$ is continuous but not necessarily smooth. Therefore, $J(x)$ is not Frechet-differentiable, but we will show that it is Gateaux-differentiable.

Minimize the functional

$$F(y) = \max_{i\in 1,2\ldots N} f_i(y) \tag{2.1}$$

where $f_i(y)$ is a real, functional defined over a subset S of a Hilbert space H. S is closed and bounded and is Frechet-differentiable ℓ times where $1 \leq \ell < \infty$. The nth Frechet differential in the direction g is denoted $\frac{\partial^n f_i(y)}{\partial g^n}$. Then we can write the following Taylor's series expansion of $f_i(y)$ in the direction $g\in H$ ($\|g\| \leq M < \infty$) and with step size α such that $(y+\alpha g)\in S$

$$f_i(y+\alpha g) = f_i(y) + \sum_{k=1}^{\ell} \frac{\alpha^k}{k!} \frac{\partial^k f_i(y)}{\partial g^k} + 0_i(|\alpha|^{\ell}) \tag{2.2}$$

where

$$\frac{0_i(|\alpha|)^{\ell}}{|\alpha|^{\ell}} \to 0 \quad \alpha \to 0$$

*This chapter follows the development of Demjanov[4] but is in Hilbert spaces.

The notation $\overline{1,N}$ designates the index set $\{1,2,\ldots N\}$.

Consider $F(y) = \max_{i\in\overline{1,N}} f_i(y)$.

We will show $F(y)$ is Gateaux-differentiable.

First we must define $R(y,\epsilon)$ and $P(y)$. Let

$$R(y,\epsilon) = \{i \mid i\in\overline{1,N},\ F(y) - f_i(y) \leq \epsilon\}$$

$$P(y) = \{i \mid i\in\overline{1,N},\ F(y) = f_i(y)\}.$$

Now since $\{\overline{1,N}\}$ is a finite set, there exists a δ, such that for all $\epsilon \leq \delta$ $R(y,\epsilon) = P(y)$.

Now for any two finite sets of real numbers A and B, with elements denoted A(i) and B(i) respectively, then

$$\max_{i\in\overline{1,N}} \{A(i) + B(i)\} \geq \max_{i\in\overline{1,N}} A(i) + \max_{i\in P} B(i) \tag{2.3}$$

where

$$P = \{i \mid i\in\overline{1,N},\ A(i) = \max_{j\in\overline{1,N}} A(j)\}$$

Proof: For any $i'\in\overline{1,N}$

$$\max_{i\in\overline{1,N}} \{A(i) + B(i)\} \geq A(i') + B(i'). \tag{2.4}$$

Now (2.4) is true for $i'\in P$, but for such i', $A(i') = \max_{i\in\overline{1,N}} A(i)$

$$\therefore \max_{i\in\overline{1,N}} \{A(i) + B(i)\} \geq \max_{i\in\overline{1,N}} A(i) + B(i') \tag{2.5}$$

and (2.5) is true for any $i'\in P$ including $\max_{i\in P} P(i)$.

$$\therefore \max_{i\in\overline{1,N}} \{A(i) + B(i)\} \geq \max_{i\in\overline{1,N}} A(i) + \max_{i\in P} B(i)$$

Q.E.D.

Now from (2.2) and (2.3) we have

$$F(y+\alpha g) = \max_{i\in 1,N} f_i(y+\alpha g) = \max_{i\in 1,N} \{f_i(y) + \sum_{k=1}^{\ell} \frac{\alpha^k}{k!} \frac{\partial^k f_i(y)}{\partial g^k} + 0_i(\alpha^\ell)\}$$

$$\geq \max_{i\in 1,N} f_i(y) + \max_{i\in P(y)} [\sum_{k=1}^{\ell} \frac{\alpha^k}{k!} \frac{\partial^k f_i(y)}{\partial g^k} + 0_i(\alpha^\ell)]. \tag{2.6}$$

On the other hand, since the f_i's are continuous, for any $\epsilon > 0$, there exists an $\alpha_1 > 0$ such that if $\alpha\in[0,\alpha_1]$ then

$$F(y+\alpha g) = \max_{i\in 1,N} f_i(y+\alpha g) = \max_{i\in R(y,\epsilon)} f_i(y+\alpha g). \tag{2.7}$$

Now ϵ must be large enough so that $R(y,\epsilon)$ contains all points which maximize $F(y)$ within the sphere with the center at y and a radius of $\|\alpha g\|$.

From the triangle inequality we have

$$F(y+\alpha g) = \max_{i\in R(y,\epsilon)} f_i(y+\alpha g) \leq \max_{i\in 1,N} f_i(y) + \max_{i\in R(y,\epsilon)} [\sum_{k=1}^{\ell} \frac{\alpha^k}{k!} \frac{\partial^k f_i(y)}{\partial g^k} + 0_i(\alpha^\ell)] \tag{2.9}$$

Thus $F(y+\alpha g)$ is between the quantities

$$F(y) + \max_{i\in P(y)} [\sum_{k=1}^{\ell} \frac{\alpha^k}{k!} \frac{\partial^k f_i(y)}{\partial g^k} + 0_i(\alpha^\ell)] \text{ and}$$

$$F(y) + \max_{i\in R(y,\epsilon)} [\sum_{k=1}^{\ell} \frac{\alpha^k}{k!} \frac{\partial^k f_i(y)}{\partial g^k} + 0_i(\alpha^\ell)].$$

As we stated previously, if ϵ is sufficiently small, i.e. $\epsilon \leq \delta$, then $R(y,\epsilon) = P(y)$. However, if we desire an ϵ that small, then the sphere with the center at y and a radius $\|\alpha g\|$ must not contain any other maxima besides those in

$P(y)$, i.e. $\alpha \leq \alpha^*$.

Therefore, when $\alpha \leq \alpha^*$ and $\epsilon \leq \delta$ then

$$F(y+\alpha g) = F(y) + \max_{i \in P(y)} \left[\sum_{k=1}^{\ell} \frac{\alpha^k}{k!} \frac{\partial^k f_i(y)}{\partial g^k} + 0_i(\alpha^\ell) \right] \tag{2.10}$$

and the directional derivative of F is

$$\lim_{\alpha \to +0} \frac{F(y+\alpha g) - F(y)}{\alpha} = \max_{i \in P(y)} \frac{\partial f_i(y)}{\partial g} . \tag{2.11}$$

3. GRADIENT METHOD

In this chapter we will present the necessary facts and methods for developing a gradient algorithm for solving the min-max problem in Hilbert spaces.

3.1. Development of Gradient Algorithm and Necessary Conditions for Solution

y^* is a local minimum of $F(y) = \max_{i \in \overline{1,N}} f_i(y)$ if there exists some $\epsilon > 0$ such that for all $y \in S$ and satisfying $\|y-y^*\| \leq \epsilon$ then $F(y^*) \leq F(y)$.*

For a gradient algorithm we assume $f_i(y)$ has a continuous Frechet derivative. We denote

$$\frac{\partial f_i(y)}{\partial g} = \langle p_i(y), g \rangle .$$

Proposition 1: A necessary condition for y^* to be a local min-max solution is that

$$\max_{i \in P(y)} \langle p_i(y), g \rangle \geq 0 \text{ all } g \in S.$$

Proof: Assume that for g'

$$\max_{i \in P(y)} \langle p_i(y), g' \rangle = k < 0.$$

*This chapter follows the development of Chapter 3 in Heller's paper[6], but is in Hilbert spaces where his is in Euclidean spaces. Most of the propositions and theorems are easily extended with the exception of Proposition 5 and Theorem 1 in this paper which corresponds with Proposition 5 in Heller's. In this case the proof in Hilbert spaces was considerably more involved.

Then there is an α sufficiently small to satisfy the requirements for equation (2.11) and where

$$|\alpha\langle p_i(y), g'\rangle| > |\ 0_i(\alpha)| \text{ for } i \epsilon P(y).$$

Therefore $F(y+\alpha g') < F(y)$ and y is not a local minimum.

Q.E.D.

Define

$$C(y) = \{\gamma | \langle p_i(y), \gamma\rangle < 0\}.$$
$$i \epsilon P(y)$$

Proposition 2: If at a point $y \epsilon S$ there exists a $\gamma \epsilon C(y)$, then there exists an $\alpha > 0$ such that

$$F(y+\alpha\gamma) < F(y).$$

The proof follows from the above.

Based on Proposition 2 a procedure can be formulated which converges to points satisfying Proposition 1. Assume y_o is an arbitrary point in S and define a sequence in $S\{y_n\}$ by

$$y_{n+1} = y_n + \alpha_n \gamma_n \tag{3.1.1}$$

where $\gamma_n \epsilon C(y_n)$, assuming $C(y_n)$ is not empty and α_n satisfies Propostion 2. If $C(y_n)$ is empty for any n, the corresponding y_n satisfies Proposition 1. The sequence $F(y_n)$ is a strictly decreasing one and is bounded by $\min_{y \epsilon S} \max_{i \epsilon 1,N} f_i(y)$. The sequence $F(y_n)$ therefore converges to a limit. As S is closed and bounded, $\{y_n\}$ must also converge to a limit point y* which satisfies Propostion 1. If the point y* did not satisfy Proposition 1, it would contradict the fact that

$F(y^*)$ is the limit of $F(y_n)$ which follows from the continuity of F. This can be summarized in the form of a proposition.

Proposition 3. The sequence of admissible points in S, defined by equation (3.1.1) converges to a point y^* which satisfies the necessary conditions for a local min-max solution.

Convergence to a point which satisfies the necessary conditions for a local min-max solution is guaranteed. It is necessary to know that a procedure eventually converges, but to be useful we must have a termination criterion which indicates when a point y_n is in the neighborhood of the solution y^*. A neighborhood termination criterion will be presented in the next section.

3.2 Termination of the Algorithm

It is desirable to terminate the sequence when a point y_n is in the neighborhood of a point satisfying Proposition 1.

Define:

$$C_k(y) = \{\gamma \in C(y) \mid \langle p_i(v), \gamma \rangle \leq -k\}$$

where $k > 0$.

Proposition 4: The set $C_k(y)$ is empty if and only if the set C(y) is empty.

Proof: If C(y) is empty, then $C_k(y)$ must be empty since it is a subset of C(y). Assume $C_k(y)$ is empty and C(y) is not.

Then there exists an $\gamma_1 \in C(y)$ such that

$$\langle p_i(y), \gamma_1 \rangle \leq -b < 0$$

but then

$$\frac{k}{b} \langle p_i(y), \gamma_1 \rangle \leq -k$$

therefore, $\frac{k}{b} \gamma_1 \epsilon C_k(y)$ which is a contradiction.

Q.E.D.

Define

$$C_k^{\epsilon}(y) = \{\gamma | \langle p_i(y), \gamma \rangle \leq -k\}$$
$$i \epsilon R(y, \epsilon).$$

Let $\|\gamma^*\|$ be the minimum element of $C_k^{\epsilon}(y)$; then as a sequence of points $\{y_n\}$ approaches the min-max solution y*, $\|\gamma^*\|$ approaches infinity. In order to prove this we use Proposition 1, i.e.,

$$\max_{i \epsilon P(y^*)} \langle p_i(y^*), g \rangle \geq 0 \text{ all } g \epsilon S. \tag{3.2.1}$$

However, in order to use this information, the point $i^* \epsilon P(y^*)$ which maximizes (3.2.1) must be included in $R(y+\alpha g, \epsilon)$. Therefore, we must show the following:

Proposition 5: For any ϵ, there exists a δ such that if

$$\alpha \leq \delta \text{ and } \|g\| = 1 \text{ then}$$

$$P(y) \subset R(y + \alpha g, \epsilon) \tag{3.2.2}$$

Proof: Since $f_i(y)$ is continuous for all i there are δ_i's such that if $\alpha_i \leq \delta_i$, then

$$|f_i(y + \alpha_i g) - f_i(y)| \leq \epsilon/2$$
$$i \epsilon \overline{1,N}$$

and

$$\left| f_i(y + \alpha_i g) - F(y) \right| \leq \epsilon/2$$

$$i \epsilon P(y)$$

since $f_i(y) = F(y)$ for $i \epsilon P(y)$.

Let $\delta' = \min_{i \epsilon P(y)} \delta_i$. Now if $\alpha' \leq \delta'$ then

$$\left| f_i(y + \alpha' g) - F(y) \right| \leq \epsilon/2$$

$$i \epsilon P(y).$$

We have shown $F(y)$ to be Gateaux-differentiable and, hence, continuous. Therefore, there is a δ_o such that if $\alpha_o \leq \delta_o$, then $\left| F(y + \alpha_o g) - F(y) \right| \leq \epsilon/2$.

Let $\delta = \min \{\delta', \delta_o\}$ and let $\alpha \leq \delta$.

Then

$$\left| f_i(y + \alpha g) - F(y + \alpha g) \right| \leq \epsilon \quad i \epsilon P(y). \tag{3.2.3}$$

But all i's satisfying (3.2.3) are elements of $R(y + \alpha g, \epsilon)$. Therefore (3.2.2) follows.

Q.E.D.

Now we are able to give the condition that indicates we are approaching the min-max solution. Simply stated, the condition is that the minimal element $\|\gamma^*\|$ of $C_k^{\epsilon}(y_n + \alpha g)$ approaches infinity as y_n approaches y^*. Or more formally, we have:

Theorem 1: For any ϵ and for any N, there is a δ such that if y^* is a local min-max solution, then if $\alpha \leq \delta$ and $\|g\| \leq 1$, then the minimum element $\|\gamma^*\|$ of $C_k^{\epsilon}(y^* + \alpha g)$ is greater than N.

Proof: Since y* is a local min-max solution, then for some i denoted i',

$$\langle p_{i'}(y^*), \gamma^* \rangle \geq 0 \qquad (3.2.4)$$

and since

$$P(y) \subset R(y + \alpha g, \varepsilon)$$

then $i' \in R(y + \alpha g, \varepsilon)$ and by the definition of γ^* we have

$$\langle p_{i'}(y^* + \alpha g), \gamma^* \rangle \leq -k \text{ for } \alpha \leq \delta .$$

Now add (3.2.4) and we get

$$\langle p_{i'}(y^* + \alpha g) - p_{i'}(y^*), \gamma^* \rangle \leq -k \qquad (3.2.5)$$

and let

$$b = p_{i'}(y^* + \alpha g) - p_{i'}(y^*).$$

Since we assumed $f_i(y)$ to have a continuous derivative, i.e. $p_{i'}(y)$ is continuous, then for any k and any N there is a δ' such that if $\|\alpha g\| < \delta'$ then

$$\|b\| \leq \frac{k}{N} .$$

Now we know from the Triangle Inequality,

$$|\langle b, \gamma^* \rangle| \leq \|b\| \, \|\gamma^*\| .$$

But from (3.2.5) $|\langle b, \gamma^* \rangle| \geq k$

$$\therefore \|b\| \, \|\gamma^*\| \geq k$$

$$\|\gamma^*\| \geq \frac{k}{\|b\|}$$

$$\|\gamma^*\| \geq N.$$

Q.E.D.

3.3 The Direction to Move

From Proposition 2 and from the fact that $C_k^\epsilon(y)$ is a subset of $C(y)$ we know that any γ which is an element of $C_k^\epsilon(y)$ will be a suitable direction which, with a small enough step, will decrease the max function. However, we may wish to travel in the direction of greatest descent. That is, if $\phi(y)$ is defined to be

$$\phi(y + \gamma) = F(y) + \max_{i \in R(y,\epsilon)} (p_i(y),\gamma) \tag{3.3.1}$$

then we want to find the γ denoted γ^* to be that direction which minimizes $\phi(y + \frac{\gamma}{\|\gamma\|})$.

$$\phi(y + \frac{\gamma^*}{\|\gamma^*\|}) = \min_{\gamma \in C_k^\epsilon(y)} \phi(y + \frac{\gamma}{\|\gamma\|}). \tag{3.3.2}$$

Theorem 2: If γ^* is the minimal element of $C_k^\epsilon(y)$ with respect to the norm, then γ^* satisfies (3.3.2).

Proof: The maximum entry of

$$\langle p_i(y),\gamma^* \rangle$$

is $-k$ where

$$i \in R(y,\epsilon).$$

Assume for an arbitrary $\gamma \in C_k^\epsilon(y)$ the maximum of $\langle p_i(y),\gamma \rangle$ is $-b$ where $i \in R(y,\epsilon)$ and $-b < -k$.

The max of $\langle p_i(y), \frac{k}{b}\gamma \rangle$ is $-k$ and $\frac{k}{b}\gamma$ is an element of $C_k^\epsilon(y)$. Therefore, we have:

$$\max_{i \in R(y,\epsilon)} \langle p_i(y), \frac{\gamma^*}{\|\gamma^*\|} \rangle = -\frac{k}{\|\gamma^*\|} \tag{3.3.3}$$

and

$$\max_{i \in R(y,\epsilon)} \langle p_i(y), \frac{\gamma}{\|\gamma\|} \rangle = \frac{-b}{\|\gamma\|} . \tag{3.3.4}$$

And by assumption that $\|\gamma^*\|$ is the minimal element of $C_k^\epsilon(y)$,

$$\|\gamma^*\| \leq \frac{k}{b} \|\gamma\|, \text{ or}$$

$$\frac{-k}{\|\gamma^*\|} \leq \frac{-b}{\|\gamma\|} . \tag{3.3.5}$$

Therefore, from (3.3.3), (3.3.4), and (3.3.5) we have

$$\max_{i \in R(y,\epsilon)} \langle p_i(y), \frac{\gamma^*}{\|\gamma^*\|} \leq \max_{i \in R(y,\epsilon)} \langle p_i(y), \frac{\gamma}{\|\gamma\|} . \tag{3.3.6}$$

From (3.3.1) we have

$$\phi(y + \frac{\gamma^*}{\|\gamma^*\|}) = F(y) + \max_{i \in R(y,\epsilon)} \langle p_i(y), \frac{\gamma^*}{\|\gamma^*\|} \tag{3.3.7}$$

and

$$\phi(y + \frac{\gamma}{\|\gamma\|}) = F(y) + \max_{i \in R(y,\epsilon)} \langle p_i(y), \frac{\gamma}{\|\gamma\|} \rangle . \tag{3.3.8}$$

From (3.3.6), (3.3.7), and (3.3.8) we have

$$\phi(y + \frac{\gamma^*}{\|\gamma^*\|}) \leq \phi(y + \frac{\gamma}{\|\gamma\|}) .$$

Put another way we have

$$\phi(y + \frac{\gamma^*}{\|\gamma^*\|}) = \min_{\gamma \in C_k^\epsilon(y)} \phi(y + \frac{\gamma}{\|\gamma\|}) .$$

Q.E.D.

3.4 Implementation of the Algorithm

We have shown the direction of steepest descent is equal to the minimum element of $C_k^{\epsilon}(y)$; however, finding the minimum $\gamma \epsilon C_k^{\epsilon}(y)$ with respect to the norm is a minimization problem in Hilbert spaces. In most problems where the Hilbert space is not the Euclidean space, this algorithm would not be suitable. However, the problem can be transformed so that the minimization is carried out in a Euclidean space.*

We wish to find $\min\langle\gamma,\gamma\rangle^{\frac{1}{2}}$ such that

$$\langle p_i(y),\gamma\rangle \leq -k \qquad i\epsilon R(y,\epsilon) \tag{3.4.1}$$

if γ' minimizes (3.4.1) it minimizes $\langle\gamma,\gamma\rangle$ subject to the same constraint.

$\therefore$ minimize $\langle\gamma,\gamma\rangle$ such that

$$\langle p_i(y),\gamma\rangle \leq -k \qquad i\epsilon R(y,\epsilon).$$

From the Kuhn-Tucker theorem[9] $\hat{\gamma}$ represents a solution to (3.4.1) if and only if a vector $\hat{u}$ exists such that

$$\Psi(\hat{\gamma},u) \leq \Psi(\hat{\gamma},\hat{u}) \leq \Psi(\gamma,\hat{u})$$

where $u \geq 0$ and

$$\Psi(\gamma,u) = \langle\gamma,\gamma\rangle + \sum_{j=1}^{K} u_j(p_j(y),\gamma\rangle + k)$$

and K is the number of elements in $R(y,\epsilon)$. This yields the Kuhn-Tucker conditions:

---

*This section deals with the Hilbert space development of the Method of Hildreth and D'Espo[11].

$$\langle p_i(y),\gamma\rangle + x_i = -k \qquad (3.4.2)$$

$i \in R(y,\epsilon)$

$$2\gamma + \sum_{j=1}^{K} u_j p_j(y) = 0 \qquad (3.4.3)$$

$$u'x = 0$$

$$u \geq 0 \quad x \geq 0 \,. \qquad (3.4.4)$$

Consequently,

$$\gamma = -\frac{1}{2} \sum_{j=1}^{K} u_j p_j(y). \qquad (3.4.5)$$

Substitute (3.4.5) into (3.4.2) and we have

$$\langle p_i(y), -\frac{1}{2} \sum_{j=1}^{K} u_j p_j(y)\rangle + x_i = -k$$

$i \in R(y,\epsilon)$

This is equivalent to

$$-\frac{1}{2} \sum_{j=1}^{K} u_j \langle p_i(y),p_j(y)\rangle + x_i = -k. \qquad (3.4.6)$$

Equation (3.4.6) along with (3.4.3) and (3.4.4) from the Kuhn-Tucker conditions of the following problem

$$\begin{aligned} &\min\ u'Gu - ku \\ &\text{with } u \geq 0 \end{aligned} \qquad (3.4.7)$$

with the matrix

$$G_{ij} = \frac{1}{4} \langle p_i(y),p_j(y)\rangle.$$

We have presented all the necessary operations to construct an algorithm employing a steepest descent search and a stopping criterion.

In Chapter 7 we will present the complete algorithm and how it should be used in conjunction with a second order approach. Therefore, we will now present the second order algorithm. It will be considered in two parts. First, we will assume that each function f_i, where

$$F(y) = \max_{i \in 1,N} f_i(y),$$

is quadratic in y and that the total number of points N in the max space is small. Next, we will consider the case where $f_i(y)$ is not quadratic and/or the number of points N is large.

4. SECOND ORDER ALGORITHM

In this chapter we will present the necessary facts and methods for developing a Newton-Raphson or second order method for solving the min-max problem in Hilbert spaces.*

4.1. The Quadratic Min-Max Problem

Consider the min-max problem

$$d^* = \min_{y \in Y} F(y) = \min_{y \in Y} \max_{i \in \overline{1,N}} f_i(y) \tag{4.1.1}$$

where the $f_i(y)$'s are quadratic functionals in the Hilbert space H, i.e.,

$$f_i(y) = a_i + \langle b_i, y \rangle + \frac{1}{2} \langle g_i(y), y \rangle$$

$g_i(y)$ is a linear functional of y and $\langle g_i(y), y \rangle$ is positive for all $y \neq 0$ and zero for $y = 0$.

The heart of the second order method is that the max function can be expressed as a function of the original variable and a vector variable. That is, $F(y) = \max_{i \in \overline{1,N}} f_i(y) = \max_{c \in C} \sum_{i=1}^{N} c_i f_i(y) = F(c,y)$. Also, F(c,y) possesses a saddle point and is equal to the minimum of F(y). That is,

$$\min_y \max_c F(c,y) = \max_c \min_y F(c,y) = \min F(y).$$

This can be expressed in the following two theorems.

---

*This chapter follows the development of Medanic[8] except in Hilbert spaces.

Theorem 1: Let the max function

$$F_m(y) = \max_{c \in C} \sum_{i=1}^{N} c_i f_i(y) \tag{4.1.2}$$

where $c = (c_1, \ldots c_N)$ and

$$\sum_{i=1}^{N} c_i = 1, \; c_i \geq 0 \; i = 1,2,\ldots N$$

and

$$F(y) = \max_{i \in \overline{1,N}} f_i(y). \tag{4.1.3}$$

Then for all $y \in H$

$$F(y) = F_m(y).$$

Proof: Let c^* maximize (4.1.2) and let c' be such that

$$\sum_{i=1}^{r} c_i' f_i(y) = \max_{i=1,N} f_i(y) = F(y).$$

Clearly

$$F_m(y) = \sum_{i=1}^{N} c_i^* f_i(y) \geq \sum_{i=1}^{N} c_i' f_i(y) = F(y) \tag{4.1.4}$$

by the definition of c^*.

On the other hand, since

$$f_i(y) \leq F(y) \quad i \in \overline{1,N}$$

it follows that

$$\sum_{i=1}^{N} c_i^* f_i(y) \leq \sum_{i=1}^{N} c_i^* F(y) = F(y).$$

Therefore,

$$F_m(y) = \sum_{i=1}^{N} c_i^* f_i(y) \leq F(y). \qquad (4.1.5)$$

From (4.1.4) and (4.1.5) we have

$$F_m(y) = F(y). \qquad \text{Q.E.D.}$$

Corollary 1. The min-max solution of the modified min-max problem

$\min_y \max_{c \in C} F(y,c)$ where

$$F(y,c) = \sum_{i=1}^{N} c_i f_i(y)$$

is equivalent to the min-max solution of the original problem (4.1.1).

Theorem 2. The modified function F(y,c) possesses a global saddle point, i.e., there exists a point (y*,c*) satisfying

$$d^* = \min_y \max_{c \in C} F(y,c) = \max_{c \in C} \min_y F(y,c).$$

Proof: Ky Fan[10] has shown that if the domains of y and c are convex and if F(y,c) is convex in y and concave in c, then

$$\min_y \max_c F(y,c) = \max_c \min_y F(y,c)$$

Hence, we must show that the above conditions are satisfied. The domain of y is not restricted and, hence, it is convex while the domain c is convex since it is defined as the intersection of two convex sets: the positive hyperquadrant $c_i \geq 0$ $i = 1,\ldots r$ and the hyperplane $\sum_{i=1}^{r} c_i = 1$.

Also, since the c_i are non-negative and $\langle g_i,(y),y\rangle$ is positive for all $y \neq 0$ then $\Sigma c_i f_i(y)$ remains quadratic and, hence, convex. And since the modified cost functional is linear in c_i $i = 1,\ldots r$, it is also concave in c.

Q.E.D.

4.2 Second Order Algorithm for the Convex Min-Max Problem

Two algorithms are possible from the above discussion, where one is imbedded in the other. The first is an algorithm which solves the quadratic min-max problem where the number of points in the maximizing space is small. Since the min-max and max-min operations can be reversed, it is necessary to find the minimum of $J(y,c)$ with respect to y in terms of c, and then maximize the resulting function of c.

This will yield the min-max solution with one maximization. Since the maximizing space c has dimensions equal to the number of points in the max set $\overline{1,N}$ then N must be kept relatively small.

If N is large and/or the functional is not quadratic, then an alternate procedure must be used.

We will consider an algorithm that makes use of second order variations.

Assume $f_i(y)$ is convex for $i \epsilon \overline{1,N}$ and $f_i(y)$ can be differentiated twice for all points.

Then from Section 2:

$$F(y + \alpha g) = \max_{i \epsilon \overline{1,N}} f_i(y + \alpha g)$$

$$F(y + \alpha g) \geq F(y) + \max_{i \epsilon P(y)} \left[\alpha \frac{\partial f_i}{\partial g} + \frac{\alpha^2}{2} \frac{\partial^2 f_i}{\partial g^2} + 0_i(\alpha^2)\right]$$

and

$$F(y + \alpha g) \leq F(y) + \max_{i \epsilon R(y,\epsilon)} \left[\alpha \frac{\partial f_i(y)}{\partial g} + \frac{\alpha^2}{2} \frac{\partial^2 f_i(y)}{\partial g^2} + 0_i(\alpha^2)\right].$$

If $f_i(y)$ is quadratic for all $i \epsilon \overline{1,N}$, then from any initial point we can immediately use an algorithm minimizing a quadratic. If, however, $f_i(y)$ is not quadratic and if the initial point is a large distance from the minimum, then higher order terms render a direction found to be no more useful than a direction found only with a gradient algorithm. And since a direction obtained with a quadratic minimization would consume more computer time than the gradient method, then the gradient method should be used.

It would be advisable to use the gradient algorithm until the step size becomes small. Therefore, assume $f_i(y)$ is quadratic or y_n is close to y^* so that third or higher order terms are negligible. Then

$$\min_{\alpha g} F(y_n + \alpha g) \geq F(y_n) + \min_{\alpha g} \max_{i \epsilon P(y)} \left[\alpha \frac{\partial f_i}{\partial g} + \frac{\alpha^2}{2} \frac{\partial^2 f_i(y)}{\partial g^2}\right]$$

and

$$\min_{\alpha g} F(y_n + \alpha g) \leq F(y_n) + \min_{\alpha g} \max_{i \epsilon R(y,\epsilon)} \left[\alpha \frac{\partial f_i}{\partial y} + \frac{\alpha^2}{2} \frac{\partial^2 f_i(y)}{\partial g^2}\right]$$

ϵ must be large enough so that $R(y,\epsilon)$ must contain all elements i' where $i' \epsilon P(y')$ and y' is within the sphere with y_n as center and $|y' - y_n|$ as radius.

The second order algorithm is as follows:

Find $\alpha^* g^*$ which minimizes

$$\max_{i \epsilon R(y,\epsilon)} \left[\frac{\partial f_i(y_n)}{\partial g} + \frac{\alpha^2}{2} \frac{\partial^2 f_i(y_n)}{\partial g^2}\right].$$

Then $y_{n+1} = y_n + \alpha^* g^*$ and $F(y_{n+1})$ should be less than $F(y_n)$. If it is not, it is because third order terms could not be neglected or ϵ was not large enough. If this is the case, find $\alpha' < 1$ such that

$$F(y_n + \alpha' \alpha^* g^*) < F(y_n).$$

Repeat the algorithm with

$$y_{n+1} \text{ and } \epsilon_{n+1} < \epsilon_n.$$

Terminate the algorithm when y_m is within δ of y^*; this occurs when $\|\alpha^* g^*\| \leq \delta$ and $\epsilon \leq \epsilon_1$ such that $R(y_m, \epsilon) = P(y_m)$.

5. THE MIN-MAX PROBLEM IN FUNCTION SPACES

We have presented the gradient and Newton-Raphson methods for solving the min-max problem in Hilbert spaces. Let us consider a particular space, namely the function space.

Given

$$F(u) = \max_{i \in \overline{1,N}} K_i(u) \tag{5.1}$$

where

$$K_i(u) = \int_0^T V(x_i(t),u(t))dt \tag{5.2}$$

and where

$$\dot{x}_i(t) = f_i(x_i,u), \text{ and } x(o) = x_o \tag{5.3}$$

Therefore, where we have used the inner product we have

$$\langle a,b \rangle = \int_0^T \sum_{i=1}^M [a_i(t)\ b_i(t)]dt.$$

5.1 The Gradient Algorithm

We must know how to calculate the gradient. The gradient of $K_i(u)$ is:

$$\text{grad } K_i = -\nabla_u H_i \text{ where} \tag{5.1.1}$$

$$H_i = -V(x_i u) + \lambda' f(x_i,u) \tag{5.1.2}$$

$$\dot{x}_i = \nabla_\lambda H \qquad x(0) = x_o \tag{5.1.3}$$

$$\dot{\lambda} = -\nabla_x H \qquad \lambda(T) = 0 \tag{5.1.4}$$

Therefore, to find the gradient of K_i at a point u, solve (5.1.3) for x(t); having this, solve (5.1.4) for $\lambda(t)$. Consequently, we can find $-\nabla_u H$.

We can now use the gradient algorithm presented in Chapter 3.

We have indicated how to find the gradient, namely by solving two sets of n differential equations where n is the order of the system and substituting the results in equation (5.1.1). The gradient is not, however, the direction to move unless there is but one point in $R(y,\epsilon)$.

We must calculate the gradient for each point in $R(y,\epsilon)$. It is then possible to find the direction to move which is the minimum element of $C_k^{\epsilon}(y)$. To find this element we minimize

$$u' G u - u \tag{5.1.5}$$

with $u \geq 0$

$$G_{ij} = \frac{1}{4} \int_0^T p_i'(t)\ p_j(t)dt$$

where p_i is the gradient of K_i associated with the ith point in $R(y,\epsilon)$. Now the direction to move is

$$\gamma = -\frac{1}{2} \sum_{j=1}^{K} u_j p_j(t) \tag{5.1.6}$$

We can then find a suitable step size and repeat the process.

Minimizing (5.1.5) is an iterative process. If a minimum is not found within a reasonable amount of time, it may be that $C_K^{\epsilon}(y)$ is empty and no value of u exists, or γ may be very large. Therefore, calculate γ and if it is greater than some N, then decrease ϵ in $C_K^{\epsilon}(y)$ or terminate because the solution has been found. If γ is less than N, continue the minimization process.

5.2 Second Order Algorithm

In Chapter 4 we stated that we must find the minimum of J(u,c) with respect to u in terms of c. We will now show how this can be done.

There are several ways in which to solve this problem. One method is as follows:

Let

$$J(u,c) = \sum_{i=1}^{N} c_i K_i(u) = \int_0^T \sum_{i=1}^{N} c_i V(x_i,u)dt.$$

This can be rewritten as

$$J(u,c) = \int_0^T \bar{V}\ (\bar{x},u,c)dt$$

where

$$\bar{x} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_N \end{bmatrix}, \quad \bar{f}(\bar{x},u) = \begin{bmatrix} f_1(x_1,u) \\ f_2(x_2,u) \\ \vdots \\ f_N(x_N,u) \end{bmatrix}$$

and $\bar{H} = -\bar{V}(\bar{x},u,c) + \bar{\lambda}'\ \bar{f}(\bar{x},u)$. If x_i is an n vector, then $\bar{x}$, and $\bar{\lambda}$ are $n \cdot N$ vectors.

If f(x,u) is linear, we can solve the Riccati equation and find u*(c), the value of u which minimizes J(u,c) in terms of c.

However, this method would consume a considerable amount of machine time and storage since we are dealing with $(N \cdot n)^2/2$ simultaneous differential equations and they must be solved many times because J(u,c) is first minimized in terms of u, then maximized in terms of c and repeated until the saddle point

is found.*

Let us consider another method for finding the minimum of J(u,c) with respect to u in terms of c.

Let

$$K_i(u) = \int_0^T V(x_i(t),\ u(t))dt. \tag{5.2.1}$$

If V is quadratic in u or if higher order terms can be dropped then

$$K_i(u) = a_i + \int_0^T (\frac{dV}{du} \cdot u(t))dt + \frac{1}{2}\int_0^T u'(t)\ \frac{d^2V}{du^2}\ u(t)dt$$

$$+ \frac{1}{2}\int_0^T \int_0^T u'(t)\ \frac{d^2V}{du(t)du(\tau)}\ u(\tau)dtd\tau. \tag{5.2.2}$$

The last two terms in (5.2.2) result from the fact that the vector $\frac{dV}{du}$ (x,u) contains u, explicitly and implicitly. Where u appears explicitly, the third term results; where u is implicit in $\frac{dV}{du}$, the fourth term appears.

We can write

$$\frac{dV}{du(t)} = \frac{\partial V}{\partial x} \cdot \frac{\partial x}{\partial u(t)} + \frac{\partial V}{\partial u(t)}$$

---

*Working concurrently but independently J. Medanic has produced an as yet unpublished paper dealing with the quadratic problem in function spaces. He has proofs similar to those in Section 4.1, and he uses the method employing the Riccati equation described above. He has considered an example where the max space N consists of two points. His paper does not present the method of finding the minimum of J(u,c) with respect to u in terms of c that is presented below, nor does he consider the gradient method, nor does he deal with the problems encountered when N is large which we will discuss in Chapter 6.

$$\frac{d^2V}{du^2(t)} = \frac{\partial^2 V}{\partial u(t)\partial x}\,\frac{\partial x}{\partial u(t)} + \frac{\partial^2 V}{\partial u^2(t)}$$

$$\frac{d^2V}{du(\tau)du(t)} = \frac{\partial x'}{\partial u(\tau)} \cdot \frac{\partial^2 V}{\partial x^2} \cdot \frac{\partial x}{\partial u(t)} + \frac{\partial x'}{\partial u(\tau)} \cdot \frac{\partial^2 V}{\partial x \partial u(t)} .$$

Now

$$J(u,c) = \sum_{i=1}^{N} c_i K_i(u).$$

Find the gradient of J(u,c) with respect to u and set it equal to zero.

$$\frac{dJ}{du} = 0$$

$$\frac{dJ}{du} = \sum_{i=1}^{N} c_i \frac{dK_i}{du} = 0 \qquad (5.2.3)$$

and

$$\frac{dK_i}{du} = \frac{dV_i}{du} + u'(t)\,\frac{d^2V_i}{du^2} + \int_0^T u'(\tau)\,\frac{d^2V_i}{du(\tau)du(t)}\,d\tau = 0 \qquad (5.2.4)$$

Putting equation (5.1.4) into (5.1.3) we get

$$\sum_{i=1}^{N} c_i\left[\frac{dV_i}{du} + u'(t)\,\frac{d^2V_i}{du^2} + \int_0^T u'(\tau)\,\frac{d^2V_i}{du(\tau)du(t)}\,d\tau\right] = 0. \qquad (5.2.5)$$

The integral equation (5.2.5) must now be solved. In general, this can be a fairly complex problem. We will now consider a specific example.

5.3 A Specific Example

We will consider the problem presented in Chapter 1.

Given a state equation

$$\dot{x} = Ax + Bu$$

and a performance index

$$J = x'(T)Qx(T) + \int_0^T u'(\tau)H\, u(\tau)dt\ .$$

Assume H is a diagonal matrix and is not a function of time. Also assume matrices A and B are constant over time, and assume that some of the elements in A are unknown, but lie within a given range. Furthermore, assume that each unknown element in A has K_ℓ values where K_ℓ denotes that the ℓth element has k values. Then if there are r such elements, we have a set of $\prod_{\ell=1}^{r} K_\ell$ points.

Let $N = \prod_{\ell=1}^{r} K_\ell$. Then the set of N points constitutes the maximizing space of the min-max problem

$$\min_u \max_{i\in\overline{1,N}} J_i(u)$$

where $J_i(u)$ is the performance index when A assumes the ith value in the set $(A_1, A_2, \ldots A_N)$.

We wish to demonstrate the gradient and second order algorithms for this problem. Now J is quadratic in u. However, assume that N is large; therefore, we do not want to generate a minimizing space of dimension N.

In order to find the gradient of K_i we must transform J so that J takes the form of (5.2).

Let

$$\dot{R}(t) = x'(t)Qx(t)$$

$$R(T) - R(0) = \int_0^T \dot{R}(t)dt = \int_0^T x'(t)Qx(t)dt$$

Therefore,

$$R(T) = x'(T)Qx(T) = x'(o)Qx(o) + \int_0^T x'(t)Qx(t)dt$$

Therefore,

$$K_i = x_i'(0)Qx_i(0) + \int_0^T [x_i'(t)Qx_i(t) + u'(t)Hu(t)].$$

Now in calculating the gradient of K_i the first term can be ignored since it is only a constant.

Therefore,

$$H_i = - x_i'(t)Qx_i(t) - u'(t)Hu(t) + \lambda'(A_i x_i + Bu(t))$$

$$\text{grad } K_i = -\nabla_u H_i = + Hu(t) - B'\lambda$$

and to find λ' we must first solve for $x(t)$ by solving

$$\dot{x} = Ax + Bu \qquad x(0) = x_o$$

then solve

$$\dot{\lambda} = 2Q\, x_i(t) - A_i'\lambda \quad \text{with } \lambda(T) = 0$$

We will now present the main computational aspects of the quadratic problem.

From the Taylor's series expansion in Chapter 2 we have:

$$\max_{i \in 1,N} J_i(u_o + h) \geq \max_{i \in 1,N} J_i(u_o) + \max_{i \in P(u_o)} [\langle \frac{dJ_i}{du}, h\rangle + \frac{1}{2}\langle h' \frac{d^2 J_i}{du^2}, h\rangle] \quad (5.3.1)$$

and

$$\max_{i\in 1,N} J_i(u_o + h) \leq \max_{i\in 1,N} J_i(u_o) + \max_{i\in R(u_o,\epsilon)} [\langle \frac{dJ_i}{du}, h\rangle + \frac{1}{2}\langle h' \frac{d^2J_i}{du^2}, h\rangle]. \tag{5.3.2}$$

Now we wish to find h to minimize

$$\max_{i\in R(u_o,\epsilon)} [\langle \frac{dJ_i}{du}(u_o), h\rangle + \frac{1}{2}\langle h' \frac{d^2J_i}{du^2}, h\rangle]$$

where ϵ is large enough to include all points in the sphere with u_o as center and $\|h\|$ as radius.

Now

$$x(t) = \Phi(t,t_o)x_o + \int_o^t \Phi(t,\tau)Bu(\tau)d\tau$$

$$\left.\frac{dJ_i}{du}\right|_{u_o} = 2x(T)Q\frac{\partial x}{\partial u} + 2u_o'(t)H$$
$$= 2x(T)Q\Phi(T,\tau)B + 2u_o'(t)H$$

$$\left.\frac{d^2J_i}{du^2}\right|_{u_o} = 2B'\Phi'(T,t)Q\Phi(T,\tau)B + 2H.$$

We have now,

$$\min_h \max_{i\in R(u_o,\epsilon)} \{\int_{t_o}^T [2x_i(T)Q\Phi_i(T,\tau)Bh + 2u_o'H\ h]d\tau$$
$$+ \int_{t_o}^T \int_{t_o}^T h'(t)B'\Phi_i(T,t)Q\Phi_i(T,\tau)Bh(\tau)dtd\tau + \int_{t_o}^T h'(x)H\ h(t)dt\}. \tag{5.3.3}$$

Now convert to a modified min-max problem.

$$\min_{h} \max_{c\in C} \sum_{i=1}^{N} c_i \Big\{ \int_{t_o}^{T} [2x_i(T)Q\Phi_i(T,\tau)Bh + 2u_o'H\,h]d\tau$$

$$+ \int_{t_o}^{T}\int_{t_o}^{T} h'(t)B'\Phi_i(T,t)\,Q\Phi_i(T,\tau)Bh(\tau)dtd\tau + \int_{t_o}^{T} h'(t)Hh(t)dt\Big\}. \qquad (5.3.4)$$

We can interchange the max and min operations and we wish to find the gradient with respect to h and set it equal to zero.

$$\sum_{i=1}^{N} c_i\Big[2x_i(T)Q\Phi_i(T,\tau)B + 2u_o'H + 2\int_{t_o}^{T} h'B'\Phi_i(T,t)Q\Phi_i(T,\tau)Bdt$$

$$+ 2h'H\Big] = 0 \qquad (5.3.5)$$

We can write this as

$$\sum_{i=1}^{N} c_i f_i(t) + Hh(t) + \int_{t_o}^{T} \sum_{i=1}^{N} c_i G_i(t)K_i(\tau)h(\tau)d\tau = 0 \qquad (5.3.6)$$

f_i is an m vector

H is an mxm matrix

G_i is a mxn matrix and

K_i is a nxm matrix.

In order to solve the integral equations (5.3.5) or (5.3.6) we must make use of the fact that if it has a solution, then h(t) must be a linear combination of all the functions in the equation. We will express this fact as equation (5.3.7). Then we will insert equation (5.3.7) into (5.3.6) and will get a set of equations which contain the vector c and a new vector b of

coefficients in (5.3.7) but the integrals no longer contain an unknown variable.

Consequently, the integrations can be calculated initially and then for a given c one can solve the matrix equation for the vector b. Then find the vector c which maximizes (5.3.10). Note that this maximization will be an iterative process, and for each new value of c, the matrix equation must be solved for b.

An element of the $\sum_{i=1}^{N} c_i f_i$ matrix is denoted $\left[\sum_{i=1}^{N} c_i f_i\right]_i$.

Let

$$h_j = -\left[\sum_{i=1}^{N} c_i f_i\right]_j / h_{jj} + \sum_{k=1}^{N}\sum_{\ell=1}^{n} c_k b_{k\ell} g_{kj\ell}. \qquad (5.3.7)$$

Put this into equation (5.3.6) and set the coefficients of $g_{pjq} = 0$.

Thus

$$c_p b_{pq} g_{pjq} + c_p g_{pjq} \sum_{s=1}^{m} \int_0^T k_{pqs}(\tau) h_s(\tau) d\tau = 0 \qquad (5.3.8)$$

$$b_{pq} + \sum_{s=1}^{m} \int_{t_o}^{T} k_{pqs} \left\{ \left[\sum_{i=1}^{N} c_i f_i\right]_s / h_{ss} + \sum_{\ell=1}^{n}\sum_{k=1}^{N} c_k b_{k\ell} g_{ks\ell} \right\} = 0 \qquad (5\ 3.9)$$

$$b_{pq} + \sum_{i=1}^{N} c_i \sum_{s=1}^{m} \int_0^T k_{pqs} [f_i]_s / h_{ss} d\tau$$

$$+ \sum_{k=1}^{N} c_k \sum_{\ell=1}^{n} b_{k\ell} \sum_{s=1}^{m} \int_0^T k_{pqs} g_{ks\ell} d\tau = 0. \qquad (5.3.10)$$

Now if b_{pq} is written as a single vector, we have

$$b + y + Ab = 0; \quad b = -[I + A]^{-1} y.$$

From examining equation (5.3.10) we have $[n \cdot N \cdot m]\ [n \cdot N+1]$ integrations to solve.

It is necessary to maximize J(u,c) with respect to c; however, it is necessary to solve the above number of integrations only once and then a new value of b and, consequently, h can be found using equations (5.3.10) and (5.3.7).

It may be noted that for a problem where the order of the system is 4, the number of inputs is 2 and $R(u_o,\epsilon) = N = 10$, $\lceil n \cdot N \cdot m \rceil \lceil n \cdot N+1 \rceil = 3280$.

However, if the method employing the Riccati equation is employed, $(n \cdot N)^2/2 = 800$ simultaneous differential equations must be solved Experience shows that it is by far easier to solve the 3280 integrations. Also, the 800 differential equations must be solved repeatedly as J is maximized with respect to c.

As stated in Chapter 4, a new point u_n is found and the process is repeated until the distance to the minimum is below some ϵ.

6. POTENTIAL DIFFICULTIES AND LIMITATIONS OF THE ALGORITHMS

In order to take a reasonably large step size with either the gradient or second order methods, ϵ in $R(u,\epsilon)$ must be large enough so that $R(u,\epsilon)$ contains all points where the max may occur within a sphere at u and a radius equal to the step size. And, clearly, near the min-max solution at least all points which maximize the original function and all other points which are very close must be considered.

In order for the algorithms to be useful, it is assumed that a relatively small number of points will solve the max problem at the min-max solution, i.e., that $P(u_o)$ is small where u_o is the min-max solution. The question remains whether this is a valid assumption.

6.1 Some Reasons Why $P(u_o)$ May be Large

Consider a min-max problem in a finite demensional space where the minimizing vector consists of one component. Experience has shown that the min-max may occur at two intersecting lines. It is not likely for more to intersect, but two is very probable.

Next consider the case in which the minimizing vector consists of two components. In this case three surfaces very likely intersect at the min-max. Continuing this reasoning, where the minimizing space is of order n, the n+1 points may maximize the function at the min-max.

Likewise, if the minimizing variable consists of an infinite number of components, then an infinite number of points may maximize the function at the min-max. Now if the max space consists of a finite number of points, then

it is very likely that a large percentage, if not all, may maximize the function at the min-max.

Now consider the problem $\dot{x} = Ax + Bu$ where some elements of A may lie within a given region and

$$J = x'(T)Qx(T) + \int_{t_o}^{T} u'(\tau)Hu(\tau)d\tau.$$

Now the max space consists of a non-countable number of points. If the minimizing variable consists of a non-countable number of elements as we have considered, then it is very likely that at the min-max solution a non-countable number of points from the max space may occur.

From this discussion it is now clear that our original assumption, that the number of points maximizing the function at the min-max is a small number, may not be reasonable.

6.2 A Numerical Example

The gradient algorithm was programmed for the following problem: find the

$$\min_{u} \max_{i \in I} J_i(u) = x'(T)Q(T) + \int_{t_o}^{T} u'Hud\tau \text{ and where } \dot{x} = Ax + Bu.$$

An example was computed where the order was 2. Q equals I, the identity matrix. There was only one input and $H = 10^{-8}$, $x_o = [1,11]$, and A varied between

$$\begin{bmatrix} -1 & -2 \\ -3 & -4 \end{bmatrix} \quad \text{and} \quad \begin{bmatrix} -2 & -3 \\ -3 & -5 \end{bmatrix}.$$

Three cases were considered: One where there were 125 points in the max space, one where 24 points were considered and one where only eight points were considered. The method used for finding the gradient was the one presented at the beginning of Section 5. The differential equations were solved using the Runge-Kutta-Gill method. The closer one approached the min-max solution, the greater degree of accuracy was necessary to find a point which decreased the function. With one level of accuracy a direction would be given, but even with an extremely small step size the function value could not be decreased. If the accuracy was then increased ten-fold, a direction could be found where the function value could be decreased from 10 to 30 percent in one iteration.

This indicates that the solution has very steep V shaped ridges as you approach the min-max solution.

The input was divided into 26 parts and in solving the differential equations each interval was further divided. In order to increase accuracy, each of the 25 intervals was divided into a greater number of sub-intervals. It may have been desirable to increase the original number of points in the control. Possibly this is a limitation of the gradient algorithm and it may be necessary to switch to the second order method.

A degree of accuracy was not obtained with the gradient method to determine what percentages of points in the max space were in $P(u^*)$ where u^* was the min-max solution.

However, one thing is clear. Even if $P(u^*)$ does not contain the entire max space or even a large part of it, $R(u_n, \epsilon)$ must be a large proportion of the max space or, preferably, all of it in order to minimize the

function in reasonable time. In the program that was constructed, $R(u_n,\epsilon)$ could only accommodate ten values. When the max space consisted of 125 points, the program took a considerable amount of computer time. The reason for this is mainly because during the maximizing step, the full 125 points had to be expanded, and when it became necessary to increase the accuracy for solving the differential equations, the program was terminated.

The program was terminated before γ^*, the minimal element of $C_k^{\epsilon}(u)$, became large. This indicates that either u_n was not close to the min-max or ϵ was not large enough so that $R(u_n,\epsilon)$ contained all the points in $P(u^*)$.

7. THE ALGORITHMS IN DETAIL

In this chapter we will present the detailed explanation of the gradient and Newton-Raphson algorithms, and how they are used together in solving the min-max problem in function spaces.

7.1 The Size of $R(u_n,\epsilon)$

As we have seen, the max function has steep gradients. Therefore, it is not desirable to pick an ϵ in $R(u_n,\epsilon)$ and find the number of points in it since it is too difficult to tell how large ϵ should be. Consequently, $R(u_n,\epsilon)$ should be as large as the following considerations allow. The algorithm which finds the minimum $\gamma\epsilon C_k^{\epsilon}(u_n)$ becomes very time consuming as the number of elements in the vector γ increases. (The number of elements in γ equals the number in $R(u_n,\epsilon)$.) There is some optimal number of elements $R(u_n,\epsilon)$ should have which depends on the total number of elements in the max space and the nature of the problem.

In the second order method we have the same problem, only to a much greater degree. There the procedure which finds the saddle point solution of $J(u,c)$ becomes exponentially more complex with each new element in the vector c. As the reader will recall, this is where either a large number of differential equations must be solved repeatedly or an even larger number of integral equations must be solved once.

The purpose of this discussion, then, is to use the gradient method with $R(u_n,\epsilon)$ having a large number of elements until one is near the solution

and then use the quadratic method with $R(u_n,\epsilon)$ being small. This is assuming that at the min-max solution P(u*) has no more elements than the number we used for $R(u_n,\epsilon)$. That is, $R(u_n,\epsilon)$ must contain at least as many elements as P(u*) and if it is a large number, then the problem for all practical purposes cannot be solved. At large distances from the min-max, it is not necessary to have a very precise figure for the gradient. It is desirable to use a crude approximation of the gradient until you are near the min-max when a greater degree of accuracy is necessary.

The gradient algorithm can be used even if J is not quadratic. The example presented in Chapters 1 and 5 is much more restricted than is necessary; however, to use the quadratic method which employs the solution of integral equations to solve the saddle point problem, it is necessary to have a very restricted problem. It is necessary that the matrix H in $J = x'(t)Qx(t) + \int_0^T u'(t)Hu(t)dt$ be diagonal. If it is not, then the set of integral equations to be solved would be much more complex.

7.2 The Gradient Algorithm

1. Choose M_1, the number of elements in $R(u_n,\epsilon)$, the initial step size α, the initial point in the minimizing space u_o, and K explained in step 5.
2. When $u = u_n$ calculate the value of the function J at all N points in the max space. Store the M_1 largest points. See Note 1 in Section 7.4.
3. Calculate the gradients for the M_1 points. See Note 2.

4. Calculate the M_1 by M_1 matrix G explained in Section 3.4 or Note 3.

5. Compute γ^* the minimum element in $C_k^{\epsilon}(u_n)$. See Note 4. If $\|\gamma^*\| \geq K$, some large number, then we are getting near the min-max. If ϵ in $R(u_n,\epsilon)$ is sufficiently small, we are at the min-max. This can be determined by noting the difference for the greatest and least value of J for the M_1 values of J in $R(u_n,\epsilon)$. This difference is equal to ϵ. If ϵ is not small enough to terminate, then decrease M_1 by one element. Now if $M_1 \leq M_2$ where M_2 is the number of points in $R(u_n,\epsilon)$ for the Newton-Raphson method, then transfer to the Newton-Raphson algorithm. If $M_1 > M_2$ then calculate a new matrix G. Note that it should be necessary only to delete the last row and column. Find a new value of γ^*. Now if $\|\gamma^*\| < K$ proceed to Step 6.

6. Let $u_{n+1} = u_n + \alpha \frac{\gamma^*}{\|\gamma^*\|}$.

7. If $F(u_{n+1}) < F(u_n)$ return to Step 2.
   If $F(u_{n+1}) \geq F(u_n)$ then let $\alpha = \frac{1}{2}\alpha$ and if $\alpha \geq \alpha_m$ return to Step 6. Observe that it is necessary to store the old value of u_n until the condition in Step 7 is satisfied.
   If α is very small, i.e. less than some α_m, the convergence is too slow. It may be that the value of γ is not accurate enough and this may result from inaccurate gradient calculations. Therefore, increase the number of points ND as explained in Note 1, Section 7.4. If this does not help, increase NO and ND and increase the accuracy of the procedure minimizing u G u - u. If none of the above helps, then switch to the Newton-Raphson algorithm.

7.3 The Newton-Raphson Algorithm

There are two conditions where the program should be transferred to the Newton-Raphson algorithm. The most desirable one is where $\|\gamma^*\|$ approaches infinity repeatedly as the number of elements in $R(u_n,\epsilon)$ is decreased until the number in $R(u_n,\epsilon)$ is quite small.

The maximum number in $R(u_n,\epsilon)$ will be denoted M_2 and should be much less than M_1 otherwise the computation time will be excessive.

The second case where the program transfers to the second order algorithm is when the gradient method produces a step which is too small to be effective.

From the experience the author has gained from working with the gradient algorithm, this will generally be the case. It may be that $\|\gamma^*\|$ never became very large and M_2 was never decreased. If this is the case, it may truly take excessive computer time to solve the problem. One can only experiment in order to know how large to make M_2, because if it is too small, you will have to decrease the step size more than what the algorithm indicates, and if it is too large, the computation will be excessive.

We will now present the Newton-Raphson algorithm employing the method of integral equations to solve the example presented in Chapters 1 and 5.

1. Choose the size of M_2 and use the first value of u from the gradient algorithm.
2. With $u = u_n$ calculate the value of the function J at all N points in the max space. Store the M_2 largest points. If the difference

between the largest and smallest value of J is less than some small quantity, then increase M_2 or terminate because it would not be possible to improve the solution.

3. Find the transition matrix at NO points in time.

4. Perform integrations indicated in Step 5.

5. Form the equation b + y + Ab = 0 and solve for b for a given c. We assume that this can be accomplished. It is not necessary that there be a unique solution.

The vector b has dimension N·n where $N = M_2$ and n is the order of the system. We will show how to construct y and A above:

Form the vector $f_i = x_i(T)Q\Phi_i(T,\tau)$ at various points of time, and where $\Phi_i(T,\tau)$ is the transition matrix associated with the ith value in $R(u_n,\epsilon)$. Form the matrices $G_i = B'\Phi_i(T,t)Q$ where G_i is a set of n·n matrices for various points of time. And form the matrices $K_i = \Phi_i(T,\tau)B$ where K_i is a set of n·m matrices for various points of time.

The notation K_{pqs} or G_{pqs} will represent the qth row and Sth column of the matrix K_p or G_p. The ℓth element of b and ℓth row of b + y + AB = 0 is obtained when $\ell = (p-1)\,n + q$ where p is the pth element in M_2 and q is the row of K_p. Therefore

$$b_\ell + \sum_{i=1}^{N} c_i \sum_{s=1}^{N} \int_0^T K_{pqs}(\tau)[f_i(\tau)]_s/H_{ss}\,d\tau$$

$$+ \sum_{k=1}^{N} c_k \sum_{r=1}^{n} b_{kr} \sum_{s=1}^{m} \int_0^T K_{pqs}(\tau)g_{ksr}(\tau)d\tau = 0 \quad .$$

The notation $[f_i(\tau)]_s$ indicates the sth element of the vector and H_{ss} is the sth diagonal element of H. Therefore

$$y_\ell = \sum_{i=1}^{N} c_i \sum_{s=1}^{m} \int_0^T K_{pqs}(\tau)[f_i(\tau)]_s/H_{ss}\,d\tau$$

and if $0 = (k-1)\,n + r$, then

$$A_{\ell o} = c_k \sum_{s=1}^{m} \int_o^T K_{pqs}(\tau) g_{ksr}(\tau)\,d\tau.$$

6. With the vector b found in Step 5 find

$$h = -\left[\sum_{i=1}^{N} c_i f_i\right]_j/H_{jj} + \sum_{k=1}^{N} \sum_{\ell=1}^{n} c_k b_{k\ell} g_{kj\ell}$$

7. Now form

$$E_i = \int_0^T \left[2x_i(t)Q\Phi_i(T,\tau)Bh + 2u_o' Hh\right]d\tau$$

$$+ \int_0^T \int_0^T h'(t)B'\Phi_i(T,t)Q\Phi_i(T,\tau)Bh(\tau)dtd\tau + \int_0^T h'(t)Hh(t)dt$$

8. Find the C which maximizes $\sum_{i=1}^{N} C_i E_i$ with $0 \le c \le 1$ and $\sum_{i=1}^{N} c_i = 1$

9. Return to Step 5 to find a new b. Repeat this loop until the vectors c and b no longer change. At this stage the saddle point has been found. Note that it is not necessary to find c at a high degree of accurancy in Step 8 at the early stages of the loop.

10. From the final value of b find h(t) as in Step 6, and form $u_{n+1} = u_n + h(t)$.

11. If $F(u_{n+1}) < F(u_n)$ proceed to Step 12. If $F(u_{n+1}) \ge F(u_n)$ then let $h = \frac{1}{2}\,h$ and return to Step 9.

12. If $\|h(t)\| \geq \epsilon_M$, the step was large and it is uncertain that we are now at the min-max. Return to Step 2.

    If $\|h(t)\| \leq \epsilon_M$, then we are at the min-max if ϵ in $R(u_n, \epsilon)$ is small. If ϵ is large then reduce ϵ and, consequently, M_2 and return to Step 2.

7.4 The Method of Calculating Various Quantities.

This section will be comprised of notes which show how to calculate the various quantities presented in the algorithms. We will deal implicitly with the example presented in Chapters 1 and 5.

Note #:

1. To calculate the function:

$$J = x_i'(T)Qx_i(T) + \int_0^T u_n'(t)Hu_n(t)dt$$

$$\dot{x}_i = A_i x + Bu_n \quad x(o) = x_o$$

We have a value of u_n at NO number of points between O and T. Further divide each interval of time by ND number of points. Use the Runge-Kutta-Gill method for solving for x(t) at each of the NO x ND points and store x(t) at each of the major NO points. Now it will be necessary to use x(t) at the NO points to calculate the gradient; however, we do not want to store x(t) for all N points in the max space. Rather, store x(t) for the first M_1 points calculated and the value of J for each of the M_1 points. Then if succeeding values of J are larger than any previous value, then

replace the new set of points x(t) with the set x(t) for which J took the least value. In this way, after finding the value of J for all N points in the max space, we will have the set of x(t) for the M_1 largest values of J. Note that it is necessary to calculate $\int_0^T u_n'(t)Hu_n(t)dt$ only once since it is not a function of A.

2. Calculation of gradients. Use the same subroutine for solving the equation $\dot{\lambda} = 2Qx_i(t) - A_i' \lambda$ with $\lambda(T) = 0$ as in Note 1. The gradient of $K_i = - Hu_n(t) - B' \lambda$.

3. Calculate matrix G. An element of G denoted G_{ij} is equal to $\frac{1}{4} \int_0^T p_i'(t)p_j(t)dt$ where $p_i(t)$ is the gradient of K_i found in Note 2.

4. Find γ^* the min of $C_k^{\epsilon}(u_n)$. Minimize u'Gu - u with $u \geq 0$ and G as in Note 3. If it is possible to find a u which minimizes the above, then $\gamma^* = - \sum_{j=1}^{k} u_j p_j(t)$. Rosenbrock's rotating coordinate system method[12] was used in this minimization. If a minimum is not found after a given number of iterations, calculate γ^* as above anyway. If $\|\gamma^*\|$ is less than some K, then continue the iterative process. If it is greater than K, return to main program.

8. CONCLUSION

This work presents the first attempt at solving the min-max problem in function spaces where the max space is larger than a few elements. Initially, it was hoped that the elimination method could be used, but as explained in the Introduction, this was not possible. Two algorithms are presented; namely, the Newton-Raphson and gradient. The gradient method was programmed and the results are presented. The gradient method was successful in the range that was anticipated, when one is far from the solution.

The best method of solving the saddle point problem which is one part of the Newton-Raphson method is the method involving integral equations.

It is quite clear that the Newton-Raphson algorithm should be used in conjunction with the gradient method. The reason for this is because if one is far from the solution, then one iteration of the Newton-Raphson method will not locate a point closer to the solution than a single iteration of the gradient algorithm; however, one iteration of the gradient method is considerably shorter than one of the second order method. However, as could be expected, convergence with the gradient method became slow near the minimum.

Also presented in this paper are some arguments why P(y) (all points from the max space which maximize the function at y) may be large compared to the total number of points in the max space. It was shown that if this happens, then the solution to the min-max problem is exceedingly difficult.

LIST OF REFERENCES

1. P. V. Kokotovic, J. B. Cruz, Jr., J. E. Heller, and P. Sannuti, "Synthesis of Optimally Sensitive Systems," Proceedings of the IEEE, Vol. 56, No. 8, pp. 1318-1324, August 1968.

2. R. A. Rohrer and M. Sobral, Jr., "Sensitivity Considerations in Optimal System Design," IEEE Trans. on Automatic Control, Vol. AC-10, pp. 43-48, January 1965.

3. J. M. Danskin, The Theory of Max-Min, Springer-Verlag, New York, 1967.

4. V. F. Demjanov, "Algorithms for Some Minimax Problems," Journal of Computer and System Sciences, Vol. 2, pp. 342-380, 1968.

5. D. M. Salmon, "Minimax Controller Design," IEEE Trans. Automatic Control, Vol. AC-13, pp. 369-376, August 1968.

6. J. E. Heller, "A Gradient Algorithm for Minimax Design," Report R-406, Coordinated Science Laboratory, University of Illinois, Urbana, Illinois, January 1969.

7. J. Medanic, "On Some Theoretical and Computational Aspects of the Minimax Problem," Report R-423, Coordinated Science Laboratory, University of Illinois, Urbana, Illinois, July 1969.

8. J. Medanic, "Solution of the Convex Minimax Problem by the Newton-Raphson Method," Proc. Eighth Annual Allerton Conf. on Circuit and System Theory, University of Illinois, October 1970.

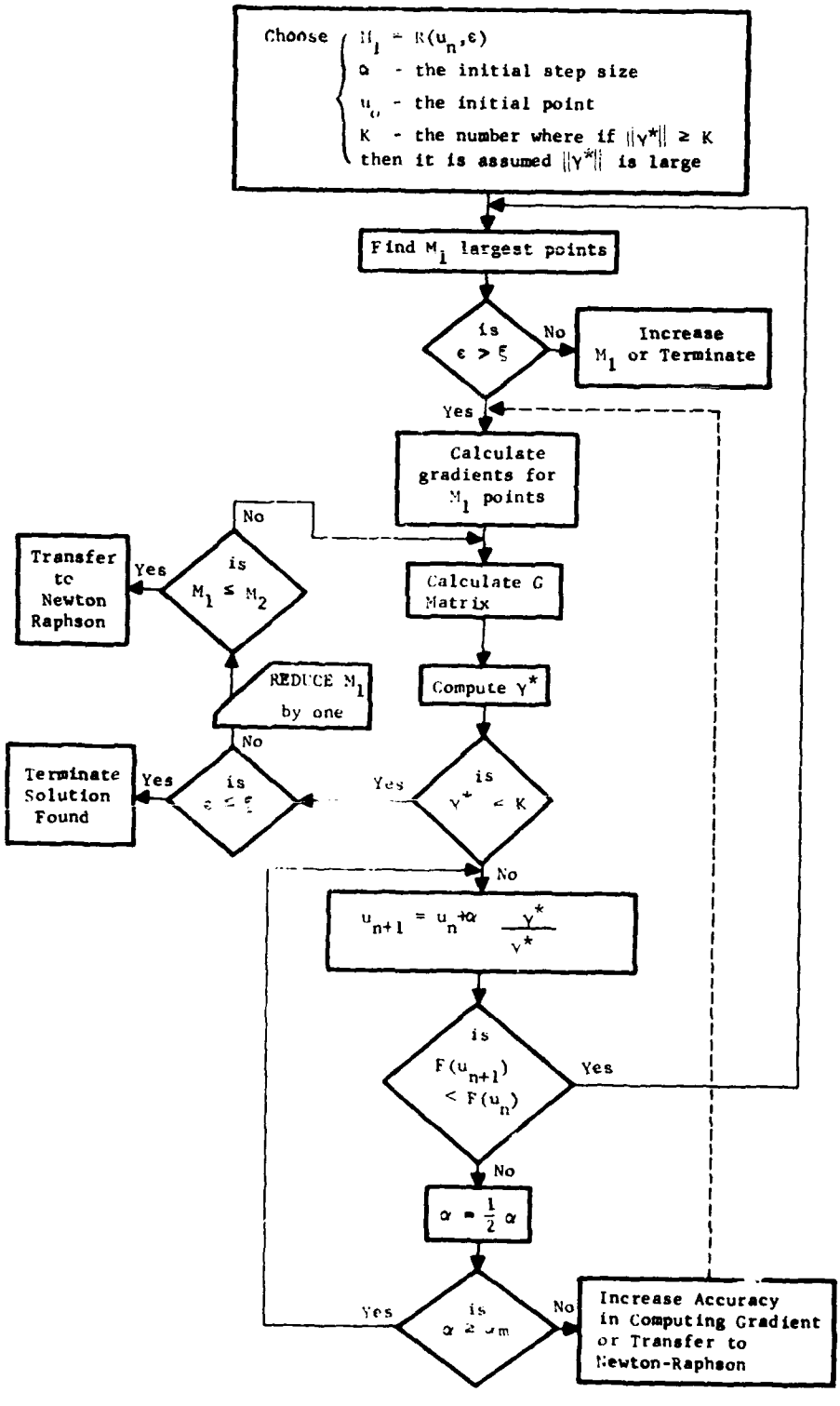
9. D. G. Luenberger, Optimization by Vector Space Method, John Wiley & Sons, Inc., New York, 1969.

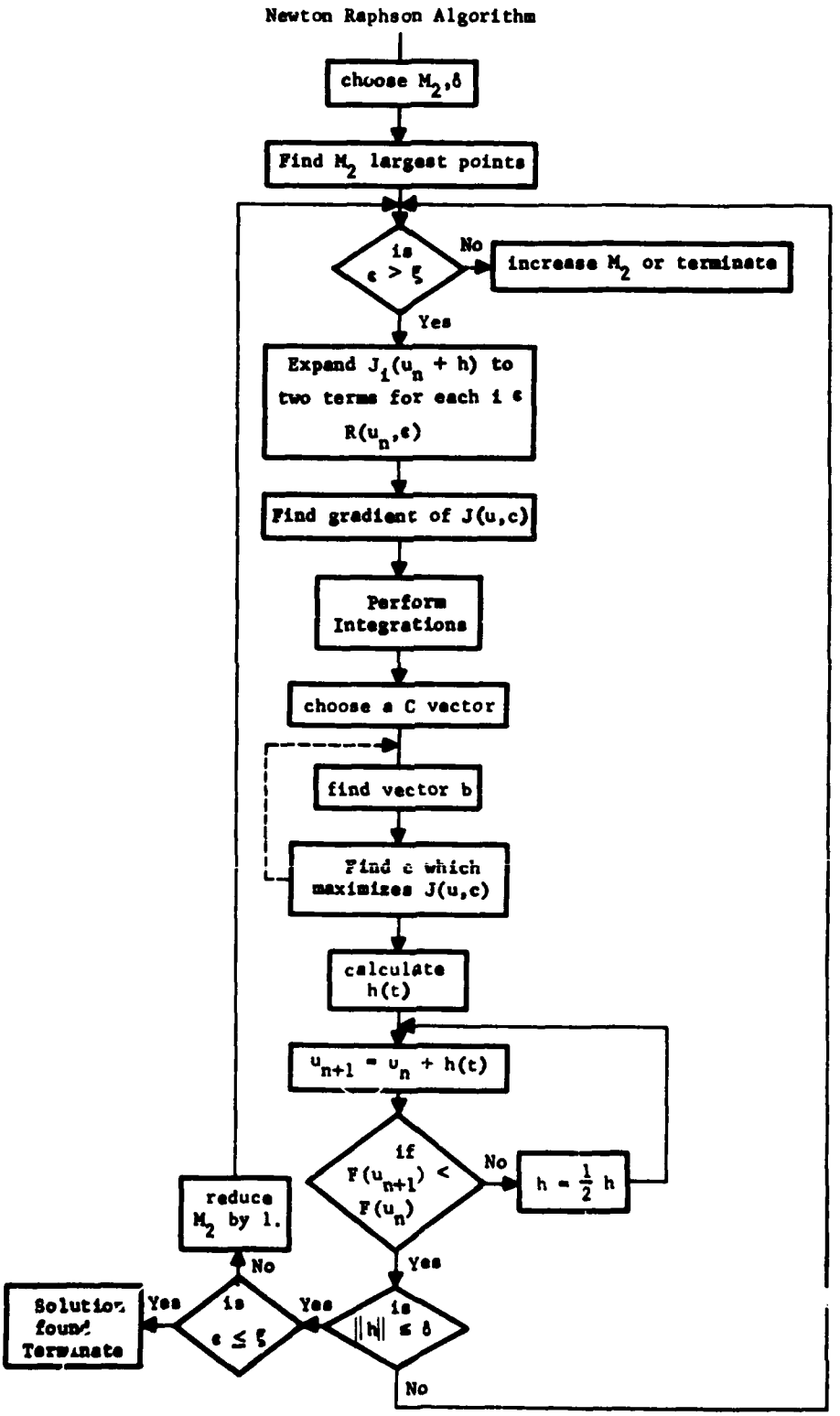
10. Fan Ky, "Minimax Theorems," Proc. Natl. Acad. Science, Vol 39, 1953.

11. H. Kunzi and W. Krelle, Nonlinear Programming, Blaisdell Publishing Co., Massachusetts, 1966.

12. C. Wood "Review of Design Optimization Techniques," IEEE Trans. on Systems Science and Cybernetics, Vol SSC-1, No. 1, pp. 14-20, November 1965.

Gradient Algorithm




Newton Raphson Algorithm
choose M_2,δ
Find M_2 largest points
is $\epsilon > \xi$
No
increase M_2 or terminate
Yes
Expand $J_1(u_n + h)$ to two terms for each i ∈ $R(u_n,\epsilon)$
Find gradient of J(u,c)
Perform Integrations
choose a C vector
find vector b
Find c which maximizes J(u,c)
calculate h(t)
$u_{n+1} = u_n + h(t)$
if $F(u_{n+1}) < F(u_n)$
No
$h = \frac{1}{2} h$
Yes
is $\|h\| \leq \delta$
Yes
No
is $\epsilon \leq \xi$
Yes
Solution found Terminate
No
reduce M_2 by 1.

VITA

Robert Wayne Hecht was born on August 26, 1943 in Chicago, Illinois. He received the Bachelor of Science degree in Electrical Engineering with honors from the University of Illinois in June, 1965. From September, 1965 to June, 1967 he was a Research Assistant in the Digital Computer Laboratory (now the Department of Computer Science) at the University of Illinois. He received his Master of Science degree in Electrical Engineering in February, 1967 from the University of Illinois. From June, 1967 to October, 1971 he was a Research Assistant in the Control Systems Group of the Coordinated Science Laboratory at the University of Illinois.

Mr. Hecht is a member of Eta Kappa Nu and Sigma Tau. He is also a member of the Institute of Electrical and Electronic Engineers.